Chapter 1

Adaptive Metropolis Sampling with Product Distributions

David H. Wolpert [1] and Chiu Fan Lee [2]

The Metropolis-Hastings (MH) algorithm is a way to sample a provided target distribution $\pi(x)$. It works by repeatedly sampling a separate proposal distribution $T(x, x')$ to generate a random walk $\{x(t)\}$. We consider a modification of the MH algorithm in which T is dynamically updated during the walk. The update at time t uses the $\{x(t' < t)\}$ to estimate the product distribution that has the least Kullback-Leibler distance to π. That estimate is the information-theoretically optimal mean-field approximation to π. We demonstrate through computer experiments that our algorithm produces samples that are superior to those of the conventional MH algorithm.

1.1 Introduction

Monte Carlo methods are a powerful tool for evaluating integrals and simulating stochastic systems. The core of any such method is an algorithm for producing a many-point IID sample of a provided **target probability distribution** $\pi(x \in X)$. Often this is done by using a Markov transition matrix $a(x, x') \equiv P(x(t+1) = x \mid x(t) = x')$. That matrix is iteratively applied starting from a randomly chosen initial x. For a properly chosen matrix, the resultant random walk asymptotically gives the desired IID sample of $\pi(x)$.

One popular method for constructing a is the Metropolis-Hastings (MH) algorithm [8, 7, 3, 4]. In this algorithm $a(x, x')$ is constructed using the ratio

[1] NASA Ames Research Center, dhw@email.arc.nasa.gov

[2] Oxford University, c.lee1@physics.ox.ac.uk

$\pi(x)/\pi(x')$, together with a **proposal distribution** $T(x, x')$. Typically T is set before the start of the Markov chain in a π-independent manner and fixed throughout the running of that chain. The rate at which the random walk produced in the associated Markov chain converges to the desired IID sample is crucially dependent on the relation between T and π however.

An important example of this is that if $T(x, x') = \pi(x)$, then the MH Markov chain is a perfect IID sampler of π (see below). Unfortunately, typically one cannot exploit this because one cannot evaluate $\pi(x)$. (Only the ratios of $\pi(x)$ values for different x can be evaluated.) However since the set $\{x(t)\}$ produced by the MH algorithm is (eventually) an IID sample of π, one can use $\{x(t)\}$ to produce a empirical estimate of π [2].

This suggests that we empirically update T during the random walk to be an increasingly accurate estimate of π. Intuitively, the idea is to try to find the density $q \in \mathcal{Q}$ that is "closest" to π and using that to update T, presuming that this will produce the most quickly converging random walk. We generically call such algorithms Adaptive Metropolis Hastings (AMH). To specify an AMH algorithm one must fix the choice of $\mathcal{Q}$, the measure of closeness, and the precise details of the resultant density estimation algorithm. One must then specify how the estimates of $\pi(x)$ are used to update $T(x, x')$.

Typically X is high-dimensional, and for the density estimation of π to work well it must be restricted to producing estimates from a relatively low-dimensional space, $\mathcal{Q}$. According, here we choose $\mathcal{Q}$ to be the set of all product distributions over X, $q(x) = \prod_i q_i(x_i)$. (Loosely speaking, this is equivalent to a mean-field approximation.) The most popular way to measure closeness between probability distributions, which we adopt here, is with the Kullback-Leibler (KL) distance, $D(p||p') \equiv -\sum_x p(x) \ln[p'(x)/p(x)]$ [1].

Given our choice of $\mathcal{Q}$, $D(q||\pi)$ is minimized for q obeying $q_i(x_i) \propto e^{-\beta E(\ln(\pi)|x_i)}$ $\forall i$ [9, 10]. Unfortunately, usually we cannot solve this coupled set of equations in closed form. However we can use sampling processes to perform an iterative search for such $D(q||\pi)$-minimizing q. To do this, we use IID samples of q to form estimates of $E(\ln(\pi) \mid x_i = s)$ for all variables x_i and associated potential values s. Those estimates are all that is needed to perform a step in a Newton's method search for $\text{argmin}_q[D(q||\pi)]$ [9, 10, 6]. Because $\mathcal{Q}$ is a product distribution, this estimation procedure scales well to large spaces X. Moreover, by construction, the estimates and the associated updates of q are parallelized, lending the algorithm to particularly fast implementation.

Another alternative, explored here, is to consider the KL distance from π to q rather than vice-versa. It can be argued that this is more correct, given the information-theory basis of KL distance [9, 10]. The product distribution minimizing this distance has the same marginals as π, i.e., obeys $q_i = \pi_i$ $\forall i$ [9, 10]. Moreover, as the random walk of the conventional MH algorithm converges to the desired IID sample of π, the i'th component of the elements of the random walk, $\{x_i(t)\}$, becomes an IID sample of $\pi_i(x_i)$. So if the number of possible x_i values is small, we can use histogramming of the random walk produced by the MH algorithm to estimate π. We don't have to run an extra process of IID sampling of q and updating it accordingly to form such an estimate.

In the next section we review the MH algorithm. Next we present our AMH algorithm. We end with experiments validating our algorithm.

1.2 Metropolis-Hastings algorithm

For a transition matrix $a(x, y)$ to preserve probability, $\sum_x a(x, y) = 1 \; \forall y$ (since $\sum_{x,z} a(x, z)\delta(y, z)$ must equal 1 for all y). Conservation of probability also means that all eigenvectors that lie on the unit simplex have eigenvalue 1, i.e., they are fixed points of a. All other eigenvectors connect points within the simplex, i.e., have the sum of their components $= 0$.[3] Moreover those eigenvectors have real eigenvalues < 1, as otherwise repeated application of a to some points in the simplex would map those points off the simplex.

So if a is full-rank and we express a distribution in terms of the eigenvectors of a, we see that running that distribution through a maps it geometrically closer to the subregion of the simplex spanned by a's fixed points. More generally, there is a fixed point which is unique if the matrix has the following properties:

- **Irreducibility:** This ensures that the probability of moving between any two states in a finite number of applications of a is non-zero. As a result the x's cannot be divided into two subsets each with its own fixed point.

- **Aperiodicity:** This means that the probability that the state is unchanged under a is non-zero.

When these conditions hold, a Markov chain based on a will map any initial point y (i.e., any initial distribution over x values, $\delta(x, y)$) to its unique fixed point distribution. So say we produce many such converged Markov chains starting from y. Then the set of last points in each chain will form a many-point IID sample of the fixed point distribution of a. Since this is true for any y, we can instead run those Markov processes one after the other, i.e., run one particularly long Markov chain. This provides our sample of π.

The MH algorithm exploits this by constructing a transition matrix with an eigenvector that is the desired distribution π. In many of the domains in which MH is used, that a is irreducible and aperiodic, so we are guaranteed of convergence to π, if π is an eigenvector. In MH a implicitly defined as follows:

1. Given current state $x(t)$, draw y from the proposal distribution $T(x(t), y)$.

2. Draw a random number r uniformly in $[0, 1]$ and update

$$x(t+1) = \begin{cases} y, & \text{if } r \leq R(x(t), y) \\ x(t), & \text{otherwise} \end{cases} \tag{1.1}$$

[3] To see this write $\sum_y a(x, y)v(y) = \alpha v(x)$, and then sum both sides over x. Since $\sum_x a(x, y) = 1$, you get $\sum_y v(y) = \alpha \sum_x v(x)$, which for $\alpha \neq 1$ implies $\sum_x v(x) = 0$.

where

$$R(x, y) = \min \left\{ 1 \ , \ \frac{\pi(y) T(y, x)}{\pi(x) T(x, y)} \right\} . \tag{1.2}$$

Note that as claimed previously, for $T = \pi$, R always equals 1, and the newly sampled point is always accepted.

Writing it out, for the MH algorithm

$$a(x(t+1), x(t)) = \min[1, \frac{T(x(t), x(t+1)) \ \pi(x(t))}{T(x(t), x(t+1)) \ \pi(x(t+1))}]. \tag{1.3}$$

For the distribution b to be a fixed point of this transition matrix it suffices to have detailed balance:

$$a(x', x) \ b^t(x) = a(x, x') \ b^t(x') \ \ \forall x, x'. \tag{1.4}$$

If both T and π are nowhere zero, this means that b must equal π. (These conditions can be weakened, but suffice for this synopsis of MH.) So for such T and π there is only one fixed point of a, and it is π, as desired.

We can generalize the foregoing by allowing T and therefore a to update stochastically in a Markovian way. To be concrete, say that for all t, $x(t+1)$ is set stochastically from T^t and $x(t)$, as in Eq. 1.3. After this T^{t+1} is set stochastically from $x(t+1)$ and T^t, and then the two-step process repeats.

All of the discussion above about geometric convergence to a fixed point still holds for Markovian evolution over (x, T). Any such fixed point $b(x, T)$ must obey

$$\begin{aligned} b(x, T) &= \int dx' dT' \ P(x, T \mid x', T') b(x', T') \\ &= \int dx' dT' \ P(T \mid x', T') P(x \mid x', T') b(x', T') \\ &= \int dx' dT' \ P(T \mid x', T') a^{T'}(x, x') b(x', T') \\ &= \int dT' \ b(T') \{ \int dx' \ P(T \mid x', T') a^{T'}(x, x') b(x' \mid T') \}. \end{aligned} \tag{1.5}$$

where $a^{T'}$ is defined in the obvious way.

For any fixed point b to provide us with an IID sample of π it will suffice if $b(x \mid T) = \pi(x) \ \forall T$. Now assume T evolves slowly, i.e., $P(T \mid x', T') \approx P(T \mid T')$. This allows us to pull it out of the integral over x'. Then since π is an eigenvector of a, our equality does indeed hold if $\forall T', \ b(x' \mid T') = \pi(x')$.

So as long as T changes slowly enough, π is still a fixed point. Moreover,, depending on how many iterations are needed for the unvarying T^t, and on the details of $P(T \mid x', T')$, it is one we reach far more quickly than under unvarying T^t. This is what AMH algorithms try to exploit.

1.3 Our AMH algorithm

1.3.1 General considerations

As mentioned above, our AMH algorithm is based on using the random walk to form increasingly accurate estimates of π and then update T^t accordingly, i.e., it is a particular choice of $P(T^{t+1} \mid x(t+1), T^t)$. There are a number of subtleties one should account for in making this choice.

In practice there is almost always substantial discrepancy between π and q, since $\mathcal{Q}$ is a small subset of the set of all possible π. This means that setting $T(x, y) = q(x)$ typically results in frequent rejections of the sample points. The usual way around this problem in conventional MH (where T is fixed before the Markov process starts) is to restrict $T(x, y)$ so that x and y must be close to one another. Intuitively, this means that once the walk finds an x with high $\pi(x)$, the y's proposed by $T(x, y)$ will also have reasonable high probability (assuming π is not too jagged). We integrate this approach into our AMH algorithm by setting $T(x, y)$ to be $q(y)$ "masked" to force y to be close to x.

Another important issue is that the earlier a point is on the random walk, the worse it serves as a sample of π. To account for this, one shouldn't form $q_i(x_i = s)$ simply as the fraction of the points for which $x_i(t) = s$. Instead we form those estimates by geometrically aging the points in forming q. This means that more recent points have more of an effect on our estimate of π. This aging has the additional advantage that it makes the evolution of τ a relatively low-dimensional Markov process, which intuitively should help speed convergence.

In [3, 7, 4] related ideas of how to exploit online-approximations of π that are generated from the random walk were explored. None of this work explicitly considers information-theoretic measures of distance (like KL distance) from the approximation to π. Nor is there any concern to "mask" the estimate of π in this work. The algorithms considered in this work also make no attempt to account for the fact that the early $x(t)$ should be discounted relative to the later ones. In addition, not using product distributions, parallelization would not be as straightforward with these alternatives schemes.

1.3.2 Details of our algorithm

Let N be the number of components of x and q^t the estimate of π at the t'th step of the walk. We consider the following algorithm:

1. Set $T^t(x, y)$ to $q^t(y)$ masked so that y and x differ in only one component:

$$T^t(x, y) \;\propto\; \delta\left(\sum_{i=1}^{N} \delta(x_i - y_i) - N + 1\right) \prod_{k=1}^{N} q_i^t(y_i)\ . \tag{1.6}$$

2. As in conventional MH, sample $[0, 1]$ uniformly to produce a r and set

$$x(t+1) = \begin{cases} y, & \text{if } r \le R^t(x(t), y) \\ x(t), & \text{otherwise} \end{cases} \tag{1.7}$$

where

$$R^t(x,y) = \min\left\{1\ ,\ \frac{\pi(y)T^t(y,x)}{\pi(x)T^t(x,y)}\right\}. \tag{1.8}$$

3. Once the chain has settled somewhat, start accumulating the sample: If $t+1 > C$, add $x(t+1)$ to the sample set D^{t+1}.

4. Once the chain has settled, periodically update q:

   If $t+1 > C$ and $\mathrm{mod}_N(t+1) = 0$, then update the set $\{q_i^t\}$:

   $$q_i^{t+1}(x_i') = \Omega_{D^{t+1}}(q_i^t(x_i'))\ \ \forall i, x_i'\ . \tag{1.9}$$

   If $t+1 > C$ and $\mathrm{mod}_N(t+1) \neq 0$, then $q_i^{t+1}(x_i') = q_i^t(x_i')\ \ \forall i$.

5. $t \leftarrow t+1$. Repeat from step 1.

In "continuous updates", $\Omega_{D^{t+1}}$ defines geometric updating of q^t by the non-negative multiplier $\alpha < 1$:

$$\begin{aligned} \Omega_{D^{t+1}}(q_i^t(x_i(t+1))) &= \alpha(q_i^t(x_i(t+1)) - 1) + 1 & (1.10)\\ \Omega_{D^{t+1}}(q_i^t(x_i' \neq x_i(t+1))) &= \alpha q_i^t(x_i'). & (1.11) \end{aligned}$$

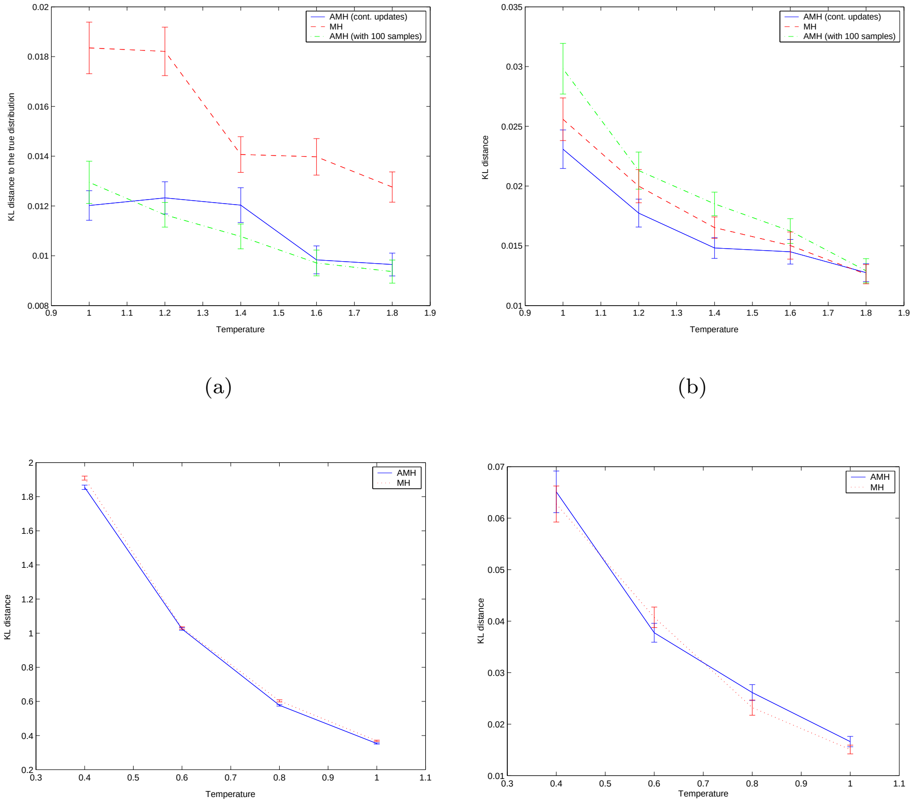
We also considered an alternative in which Ω^{t+1} is set in a manner similar to that used in [7]. Under this alternative, "resampling updates", one uniformly randomly samples D^{t+1} to form a sample S of L points. One then sets

$$\Omega_{D^{t+1}}(q_i^t(x_i')) = \frac{L_{x_i'}}{L} \tag{1.12}$$

where $L_{x_i'}$ is the number of $x \in S$ such that $x_i = x_i'$.

1.4 Experiments

Currently there is no consensus on how to quantify "how close" a set $\{x(t)\}$ is to an IID sample of π. One approach is to input the sample into a density estimation algorithm [2]. One can then use KL distance from that estimated distribution to π as the desired quantification. This can be problematic in high-dimensional spaces though, where the choice of density estimation algorithm would be crucial. However say we have a contractive mapping $F : x \in X \to y \in Y$ where Y is a low-dimensional space that captures those aspects of X that are of most interest. We can apply F to the random walk to produce its image in Y. Next one can apply something as simple and (relatively) unobjectionable as histogramming to do the density estimation in Y. We can then use KL distance between that histogram and $F(\pi)$ as the desired quantification of how good our transition matrix is. This the approach we took here.



(a)

(b)

(c)

(d)

Figure 1.1: KL accuracy of samples, for energy and for the number of x_i that equal 1, for the spin glass problem ((a) and (b) respectively) and for the 3-SAT problem.

All of our experiments had $C = 10000$ and $\alpha = 0.99$. All results reported are for 100 separate 20000-step Makrov chains. In our spin glass experiments the states for each spin (i.e., each x_i) are $\{-1, 1\}$, and -ln($\pi(x)$) is the "energy"

$$E(x) = \frac{1}{2} \sum_{<i,j>} J_{ij} x_i x_j + \sum_i h_i x_i \tag{1.13}$$

where the sum is over all neighboring pairs on a two-dimensional rectangular grid. The constants $\{J_{ij}, h_i\}$ are generated uniformly from the interval $[-1, 1]$.

We also considered a 3-SAT-problem with 20 variables and 85 clauses (so the problem is hard as 85/20=4.25). Negations in clauses and inclusion in a clause are randomly generated. The parameters in the simulation are the following;$C =$ 10000, and $\alpha = 0.99$. Here the energy is -1 times the number of violated clauses.

The figure shows that in these experiments, AMH outperforms conventional MH, and continuous updating gave better results than resampling updating.

1.5 Other uses of q

The q produced by AMH has many uses beyond improved sampling. It can be used as an estimate of the marginals of π, i.e., as an estimate of the optimal mean-field approximation to π. Because they are product distributions, the successive q^t can also be used as the control settings in adaptive distributed control [10, 5]. (In this application $\{x(t)\}$ is the sequence of control variable states and π is log of the objective function.) It's being a product distribution also means that the final q can be used to help find the bounded rational equilibria of a non-cooperative game with shared utility functions [9].

Acknowledgements: We thank Bill Macready for stimulating discussion. C.F.L. is a Junior Research Fellow at University College (Oxford) and he thanks NASA and NSERC (Canada) for finanical support.

Bibliography

[1] Cover, T., and J. Thomas, *Elements of Information Theory*, Wiley-Interscience New York (1991).

[2] Duda, R. O., P. E. Hart, and D. G. Stork, *Pattern Classification (2nd ed.)*, Wiley and Sons (2000).

[3] Gasemyr, Jorund, "On an adaptive version of the metropolis-hastings algorithm", *Scandinavian Journal of Statistics* **30** (2003), 159–173.

[4] J.N.Corcoran, and U. Schnieder, "Pseudo-perfect and adaptive variants of the metropolis-hasting algorithm", unpublished (2004).

[5] Lee, C. Fan, and D. H. Wolpert, "Product distribution theory for control of multi-agent systems", Submitted to AAMAS 04 (2004).

[6] Macready, W., and D. H. Wolpert, "Distributed optimization", Submitted to ICCS04 (2004).

[7] Sims, Christopher, "Adaptive metropolis-hastings sampling", unpublished (1998).

[8] West, Mike, "Mixture models, monte carlo, bayesian updating and dynamic models", *Computing Science and Statistics* **24** (1993), 325–333.

[9] Wolpert, D. H., "Information theory — the bridge connecting bounded rational game theory and statistical physics", *Complex Engineering Systems* (A. M. D. Braha and Y. Bar-Yam eds.), (2004).

[10] Wolpert, D. H., and S. Bieniawski, "Theory of distributed control using product distributions", *Proceedings of CDC04*, (2004).